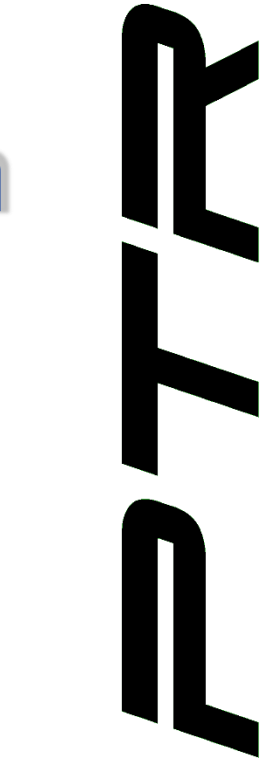
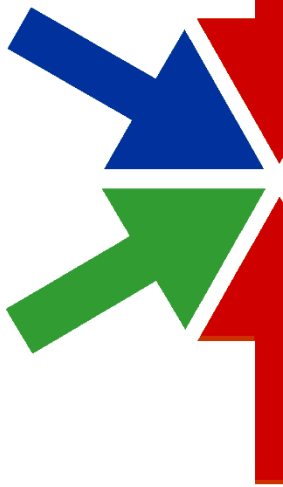
Implementing Controls with Bluetooth Smart in Android

Mike Anderson
mike@theptrgroup.com
http://www.theptrgroup.com





What We'll Talk About

- Bluetooth variants
- Bluetooth's role in the IoT
- Sample devices
- Bluetooth support in Android
- Dealing with broadcast devices
- Connecting to a Bluetooth Smart device
- Reading and writing to the device
- Summary

Classic Bluetooth

- Bluetooth was originally introduced by Ericsson in 1994 as a wireless alternative to serial ports
  - The serial port profile is still in wide usage
- Operating in the 2.4 GHz ISM band, Bluetooth has gone through multiple revisions
  - Occupies 79 channels with each channel having a 1 MHz bandwidth and a maximum transfer rate of 24 Mbit/s
- IEEE originally standardized as IEEE 802.15.1 but they have now abandoned the specification and it is now owned by the Bluetooth SIG
  - Current version is V4.2
- Communications is a master/slave style with each master capable of communicating with up to 7 devices at a time
  - Requires pairing where devices exchange UUIDs and maintain a PIN for access control

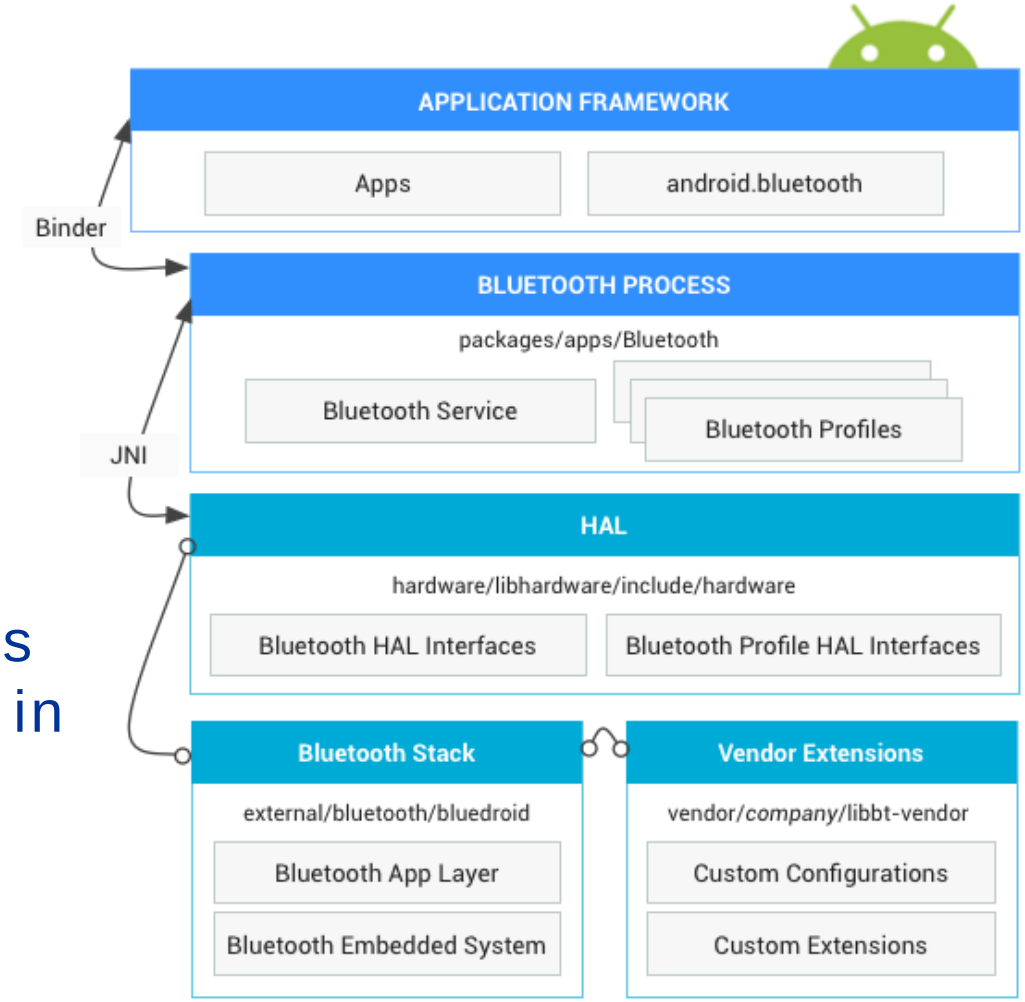
Source: bluetooth.org

Android's Problems with Bluetooth

- Up until Android 4.2, the Android Bluetooth stack was based on BlueZ from Linux
  - Covered under GPLv2
  - Very stable
- Because of Google's desire for an ASL license everywhere, BlueZ was dropped and BlueDroid was introduced
  - Almost completely written and maintained by Broadcom with some help from Google
  - Started Bluetooth stack essentially from scratch
  - Bluetooth stability has suffered greatly

BlueDroid Architecture in Android

- The architecture of the new Bluetooth Android solution doesn't look that much different from the BlueZ approach
- Vendors may make modifications to the HAL and device drivers for specific chipset implementation differences
- BlueDroid sources are still in the AOSP tree in the hardware subdirectory



Source: android.com

When is Bluetooth not Bluetooth?

- Bluetooth Low Energy (BLE), a.k.a. Bluetooth Smart or Bluetooth 4.x, was originally known as Wibree
  - Introduced by Nokia in 2006
  - Merged with core Bluetooth standard in 2010

Source: nokia.com

- Other than the name and the frequency range, Bluetooth Smart has little to do with classic Bluetooth
  - Different modulation technique
  - Not compatible with classic Bluetooth devices
  - Different application profiles and concept of operations
  - No pairing required
- Bluetooth specification allows for simultaneous classic and low energy communications through a single antenna
  - But, that's about all they have in common

Bluetooth Smart Branding

- To further confuse things, the Bluetooth SIG introduced the Bluetooth Smart logo to help "clarify" compatibility
- Bluetooth Smart Ready indicates that the device supports dual-mode - classic and low-energy operation
  - The first smartphone to implement BLE was the iPhone 4S
  - The Bluetooth SIG predicts that 90% of Bluetooth-enabled smartphones will support Bluetooth Smart by 2018
- Bluetooth Smart indicates that the device is low-energy only
  - Targets applications where the device is expected to run for months to years on a coin cell
  - Small size and low-cost sensors
  - These are part of the infamous Internet of Things (IoT) ☺

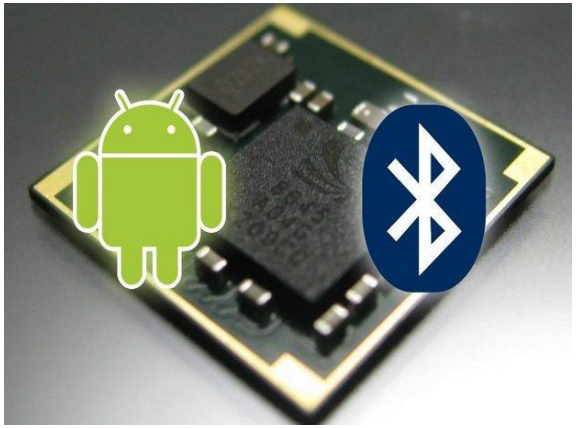
Source: bluetooth.org

Characteristics of BT Smart

- The theoretical range of BT Smart devices is > 100m
  - OTA data rate is 1 Mbit/s
  - Application throughput is 0.27 Mbit/s
- Uses AES-128 with counter mode CBC-MAC encryption
- Latency from a non-connected state is ~6ms compared to 100ms for classic BT
- Power consumption is 0.01-0.5W depending on the use case
  - Peak current consumption < 15mA
- There is a gateway capability for linking BT Smart devices to the Internet
  - Linux or Android can be used in the gateway service
- These features of BT Smart make it ideally suited to IoT type applications

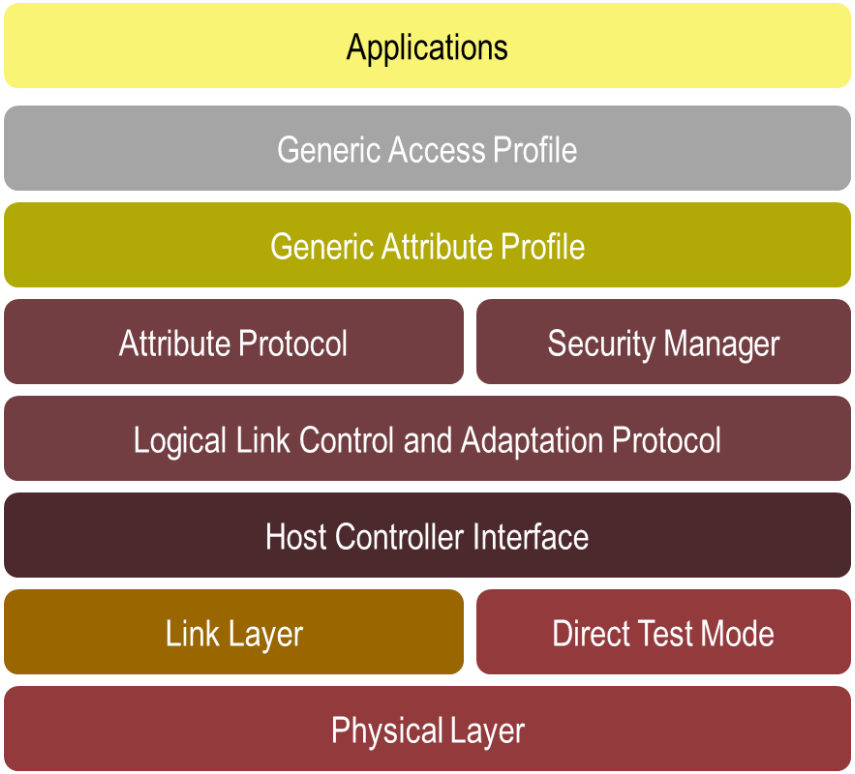
Silicon Implementation

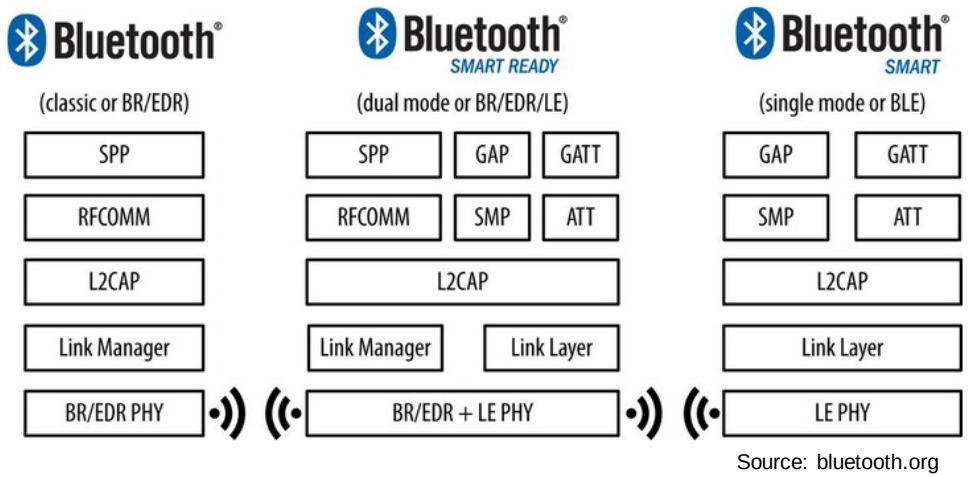
- A number of manufacturers have introduced BT Smart chipsets
  - Often based on SDR implementations so they are easily upgraded
  - Examples include TI, Cambridge Silicon Radio, Nordic Semiconductor, STMicroelectronics and more
- The Bluetooth SIG maintains a list of compatible BT Smart devices at
  - http://www.bluetooth.com/Pages/Bluetooth-Smart-Devices-List.aspx



Source: wirelessintegrated.com

Overall Bluetooth 4.0 Architecture





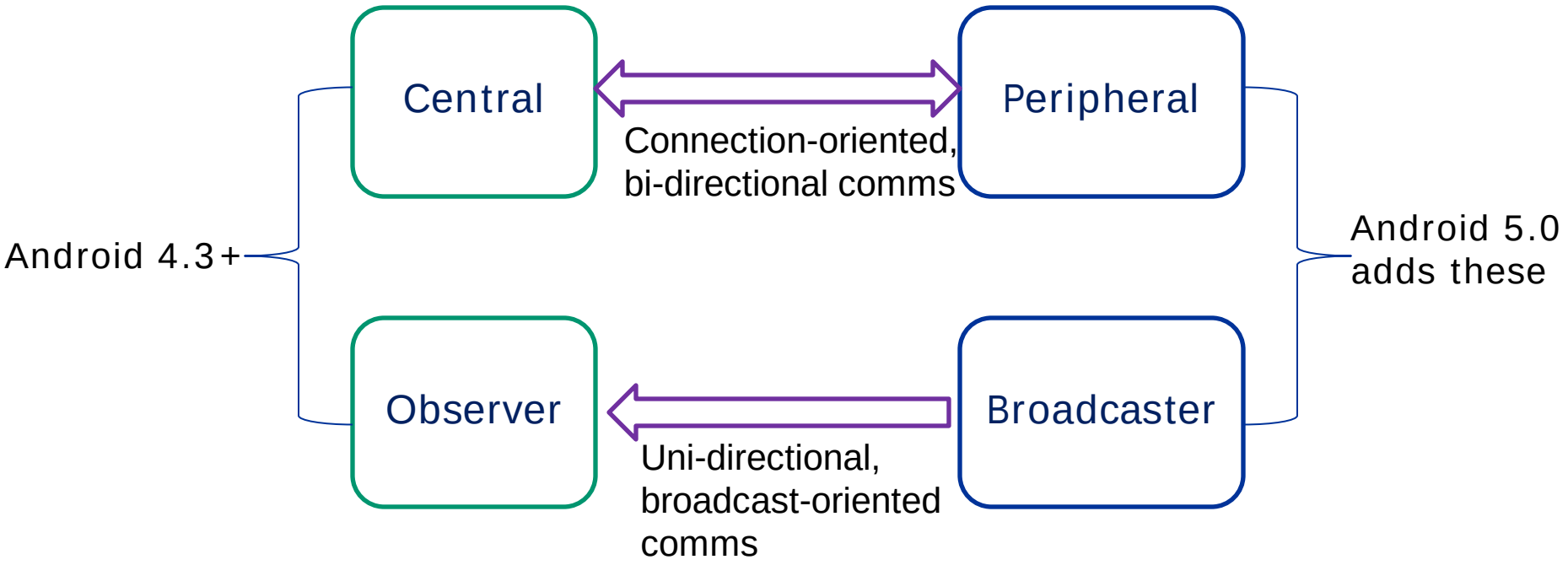
Source: bluetooth.org

Bluetooth LE Support

- BLE support was introduced in Android 4.3
- Due to the newness of BlueDroid, the BLE was finicky and tended to fail a lot
  - Difficult to even get a demo working
- Huge number of commits for BlueDroid from 4.4 to 5.0
  - Bluetooth stability in general has improved significantly
  - New BLE APIs and new Bluetooth device manager was introduced in 5.0
    - This means there's a way to do it in 4.3/4.4 and a different way in 5.x
      - 5.x can run 4.3/4.4 code, however

BLE Roles

- BLE supports 4 major roles:



Scans for devices that are advertising their presence

Advertise their presence to the outside world

New Profiles

- BT Smart introduces a hoard of new profiles for specific applications
  - All derived from the generic attribute (GATT) profile
    - Profiles are comprised of a series of key/value pairs
- New profiles include:
  - Health care profiles
    - Blood pressure (BLP), Health Thermometer (HTP), Glucose (GLP) and continuous glucose monitor (CGMP) profiles
  - Sports and fitness profiles
    - Body composition service (BSC), bicycle/exercise bike cadence and wheel speed (CSCP), cycling power (CPP), heart rate (HRP), location and navigation (LNP), running speed and cadence (RSCP) and weight scale (WSP) profiles

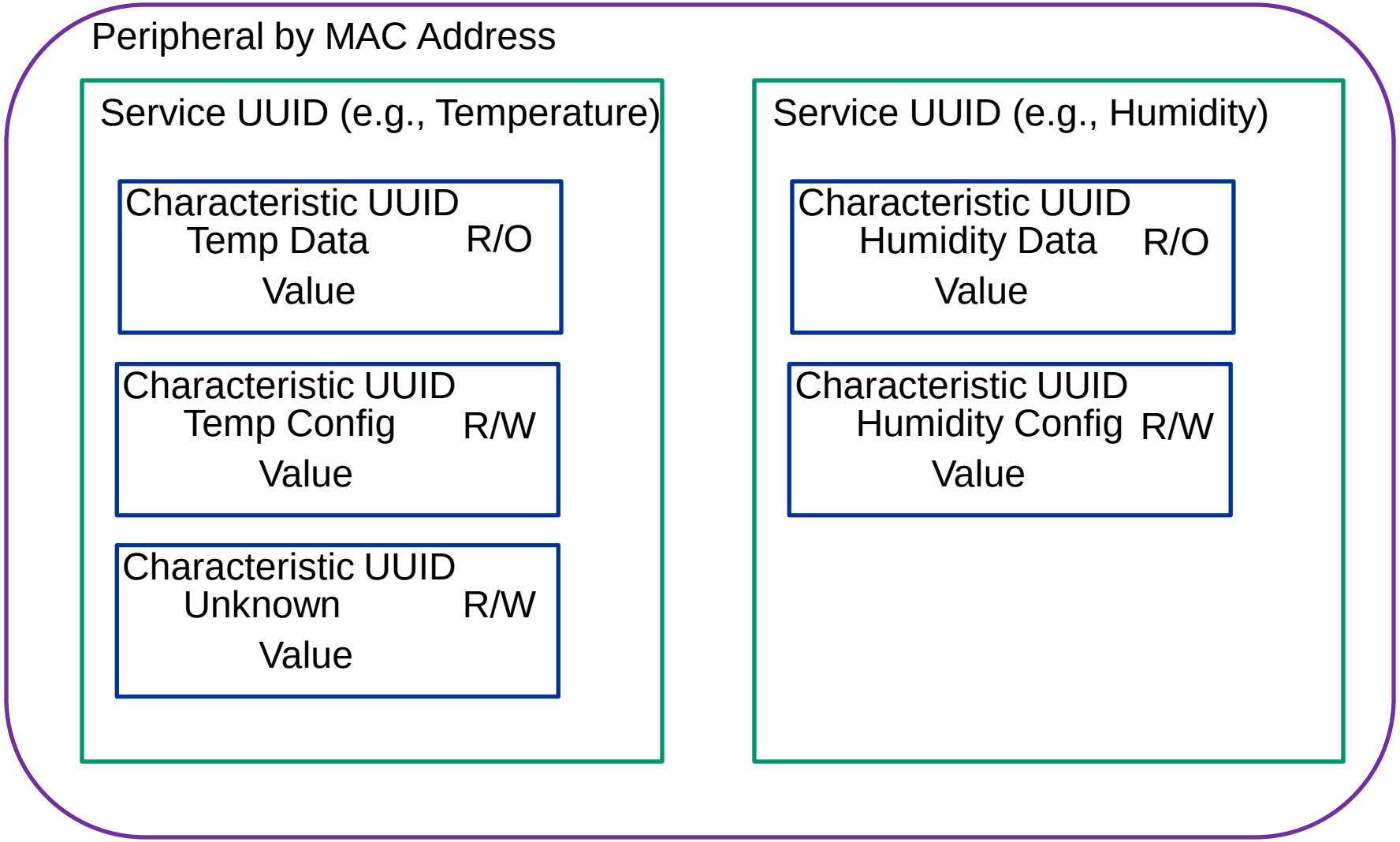
New Profiles #2

- Internet connectivity
  - Internet Protocol Support Profile (IPSP)
  - Supports IPv6/6LoWPAN or BT Smart Gateways
- Generic Sensors
  - Environmental sensing profile (ESP)
  - User data service (UDS)
- HID Connectivity
  - HID over GATT profile (HOGP)
- Proximity sensing
  - Find me profile (FMP)
  - Proximity profile (PXP)
- Alert and time profiles
  - Allows for notifications such as incoming phone calls or text messages for devices such as smartwatches

General GATT Peripheral View

- A given profile is contains services
  - Either by full 128-bit UUID or by BT SIG-defined 16-bit assigned numbers
- Services are made up of characteristics
  - These may be read/write or read only
- Characteristics then contain properties (called descriptors) or values
  - There may be one or more of these for a given service
  - A descriptor may tell you what units are being used for a particular value

GATT Peripheral View



Using UUIDs

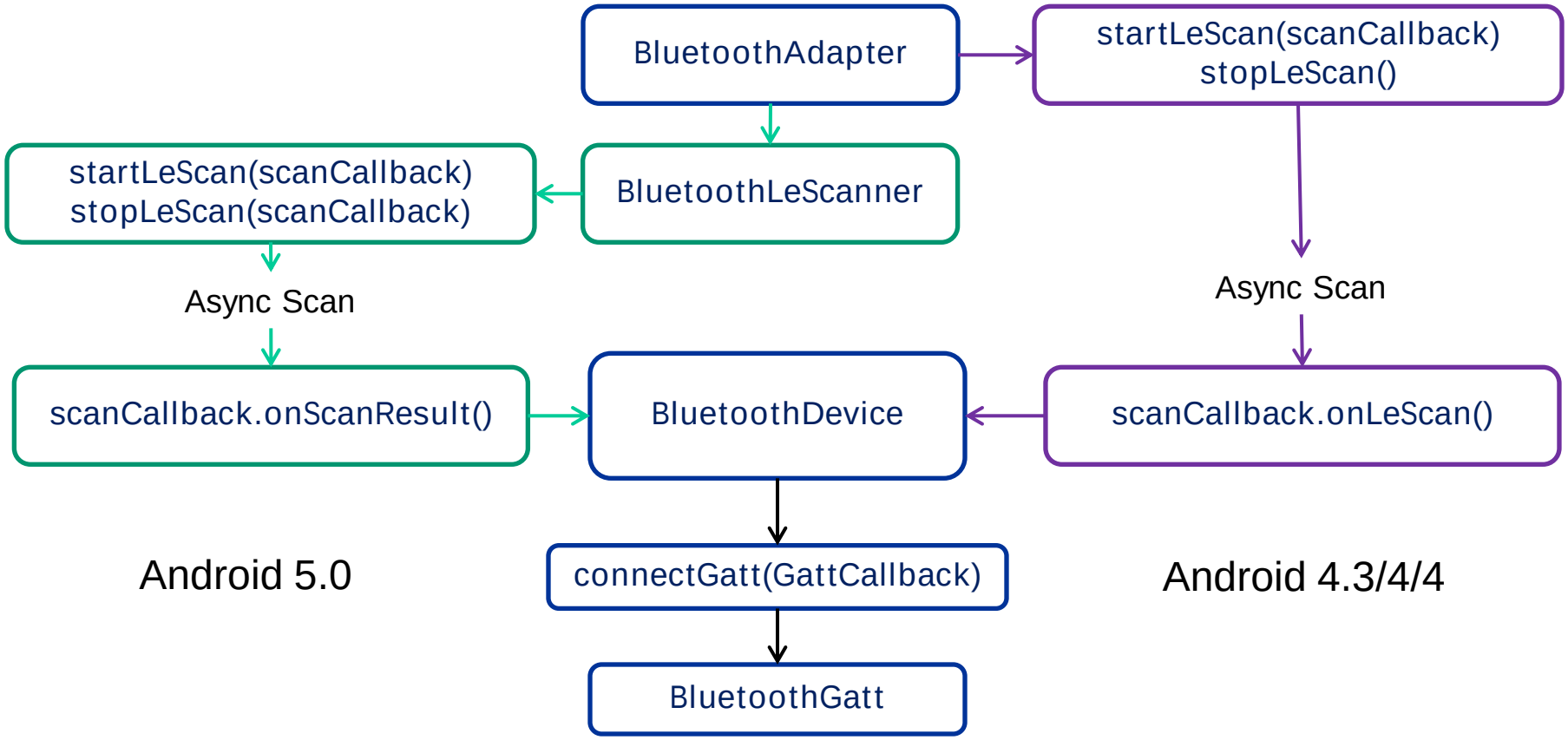
- For your Android code, you will always need to pass the complete 128-bit UUID
  - Even the 16-bit shorthand is part of the larger common base UUID
- There are Android APIs for constructing the UUID
- From a scan, the characteristic UUID is:
  - F000aa00-0451-4000-b000-000000000000
  - In this case the “aa00” would be the 16-bit assigned UUID from the BT SIG
- Predefined UUIDs can be found here: https://www.bluetooth.org/en-us/specification/assigned-numbers

Improvements with Android 5.0

- The 4.3+ implementation of BLE was pretty limited
  - Limited ability to filter data sources
    - 5.0 can now filter on any advertised field
  - No mechanism for parsing scan results
    - New framework automatically parses the advertisement
  - Only able to receive one advertisement per device
    - Now you can see all advertisements form a given device at the same time
  - Only active scanning
    - Now supports batching of scan reports to save battery
- Lollipop now supports advertisement and the creation of a GATT server to make Android a peripheral
  - Check **`BluetoothAdapter.isMultipleAdvertisementSupported()`** return to see if your device has the capability enabled

GATT Discovery

- Discovery varies slightly between 4.3/4.4 and 5.0



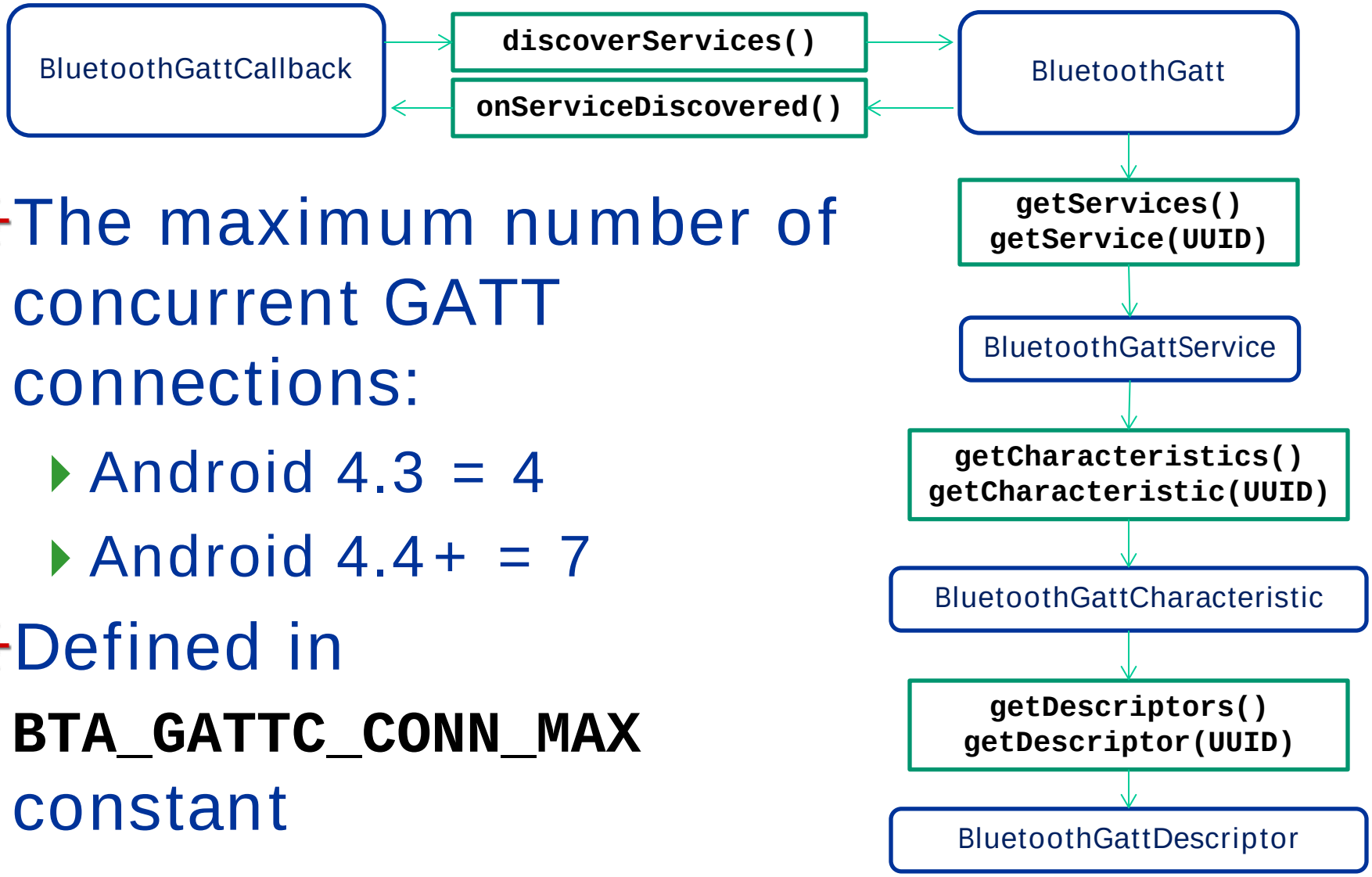
Broadcast Operation

- As its name implies, a device running in broadcast mode is simply blasting out data in its advertisements
  - Limited to 31 bytes of payload per ad with the potential of 2 ads yielding 62 bytes of payload
- Broadcast devices send blindly not knowing if there's anyone there to receive
- The observers need to scan for the devices to find what is being advertised
- Essentially, once you've scanned for the device and read and parsed its ad data, you're finished

Peripheral Operation

- Once you discover a device that you want to connect with, you'll need to take a few extra steps
- You take the **`BluetoothDevice`** instance and call **`connectGatt(callback)`** method
  - This gives a reference to a **`BluetoothGatt`** instance
    - You're now connected to the device and you can use the **`discoverServices()`** method to start enumerating the services from the peripheral

Service/Characteristic Discovery



- The maximum number of concurrent GATT connections:
  - Android 4.3 = 4
  - Android 4.4+ = 7
- Defined in **`BTA_GATTC_CONN_MAX`** constant

Notifications vs. Polling

- Polling would require the Android device to continuously run and would likely kill our battery quickly
- Fortunately, BLE supports having the GATT server push characteristic changes via notifications to Android
- The maximum number of active notifications has changed over the versions (**`BTA_GATTC_NOTIF_REG_MAX`** constant)
  - 4.3 = 4 max
  - 4.4 = 7 max
  - 5.0+ = 15 max

Reading/Writing Characteristics

- Once we have the characteristic reference of type **`BluetoothGattCharacteristic`**, we can read and write the values
- There are a series of public methods for manipulation the characteristics including both a **`getValue()`** and a **`setValue()`**
- There are also methods for getting the UUID
- Look at the Android documentation for a complete list of the API:
  - https://developer.android.com/reference/android/bluetooth/BluetoothGattCharacteristic.html

Summary

- We've taken a whirlwind tour of the new Bluetooth Low Energy operation in Android
- BLE is designed to be able to run for months to years on a coin cell battery
  - Sensors can be very long-lived with BLE
  - The internet connectivity profile provides a gateway service to the Internet
    - This is a good match for IoT applications
- Even though BLE was introduced in Android 4.3, problems with the BlueDroid stack and incompleteness of the APIs mean you should probably only use Android 5.0+ for BLE applications
- With Lollipop, Android now has the ability to be both a client/observer and a peripheral/ broadcaster
  - The Android handset can be either a display for captured sensor data or a source for sensor data
- Time for a demo…